10-2004

Compliant Encryption of JPEG2000 Codestreams

Yongdong WU
Institute for Infocomm Research

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

DOI: https://doi.org/10.1109/ICIP.2004.1421854

COMPLIANT ENCRYPTION OF JPEG2000 CODESTREAMS

Yongdong Wu and Robert H. Deng

Institute for Infocomm Research
21, Heng Mui Keng Terrace, Singapore, 119613
{wydong, deng}@i2r.a-star.edu.sg

ABSTRACT

This paper presents a compliant encryption method for JPEG 2000 codestreams such that the encryption process does not introduce superfluous JPEG2000 markers in the protected codestream, i.e., the protected codestream preserves the syntax of the original codestream. The proposed encryption method works with any standard ciphers, incurs no storage overhead, introduces negligible computational cost and maintains all the desirable properties of the original JPEG 2000 codestream such as error resilience and scalability.

1. INTRODUCTION

JPEG2000 [1] is the latest international still image compression standard. JPEG2000 Security (or JPSEC in short) is Part 8 of the JPEG2000 standard. One major technical issue in JPSEC is encryption (e.g., for codestream access control and confidentiality protection). A JPEG2000 codestream is composed of markers and data packets. The markers, with values restricted to the interval [0xFF90, 0xFFFF], are used to delimit various logical units of the codestream, facilitates random access, and maintains synchronization in the event of error-prone transmission. The packets carry the content bitstreams whose codewords (i.e., two contiguous bytes) are not in the interval [0xFF90, 0xFFFF]. Since the output of a good cipher appears "random", straightforward application of a cipher to encrypt codestream packets is bound to produce encrypted packets which include superfluous markers. Such markers will cause potentially serious decoding problems (such as loss of codestream synchronization and erroneous or faulty image transcoding). To overcome the superfluous marker problem, the encryption method must be JPEG2000 codestream syntax compliant. Such a compliant encryption method does not introduce superfluous markers in the encrypted packets and maintains all the desirable properties of the original codestreams.

Zeng *et al.* [2][3] selectively shuffle MPEG streams using shuffling tables so as to maintain syntax compliance. This shuffling method was generalized to index mapping in [4]. However, the schemes in [2]-[4] are not applicable to JPEG2000 codestream because the shuffling tables are too large to be implemented in practice.

A syntax-aware encryption scheme was proposed by Wu and Mao [4]. This scheme applies bit stuffing in conjunction with selective bitstream encryption to overcome the superfluous marker problem. However, bit stuffing not only incurs some storage overhead, but also leads to additional work for updating packet headers.

Conan *et al.* [5] described a technique which selectively encrypt JPEG2000 codestreams in order to generate compliant encrypted codestreams. In this scheme, if any byte, say X, has a value less than 0xF0, the four LSBs (Least Significant Bits) of X are encrypted with a block cipher. Clearly, the security of this simple scheme is weak.

Wu *et al.* [6] introduced a word-level encryption scheme employing a stream cipher without scarifying the strength of security. It encrypts packets in a codestream using arithmetic addition-module operation, rather than the standard XOR-module operation. However, the scheme produces only partial compliant protected codestreams.

Canon inc. proposed another word-level scheme [7] which encrypts a word recursively until the ciphertext is compliant. This scheme not only has to check the value of the current encrypted word, but also its proceeding byte and succeeding byte. Ma *et al* [8] pointed out that Canon's proposal was not reversible, i.e., portion of a plaintext can not be recovered from the corresponding ciphertext.

Wu and Ma [9] proposed two packet-level encryption schemes based on stream ciphers and block ciphers, respectively. They showed that the two schemes protect 99% of a codestream. However, the schemes are not able to regain synchronization when a byte is changed into 0xFF or when part of the packet is lost.

This paper presents a codeblock-level compliant scheme which independently encrypts Codeblock Contribution to Packets (CCPs) using standard ciphers. Our scheme provides full protection of codestreams and allows direct manipulation of the codestreams at the granularity of codeblocks. In addition, the protected codestreams inherit all the desirable properties of the original JPEG2000 codestreams, e.g., error resilience and scalability.

The rest of this paper is organized as follows. Section 2 briefs the structure of JPEG2000 packets. Section 3 describes our proposed compliant encryption scheme. Section 4 analyzes the performance of the proposed scheme. Simulation results in Section 5 demonstrate the efficiency of the proposed scheme. A conclusion is drawn in Section 6.

2. JPEG2000 PACKET STRUCTURE

In JPEG2000 standard [1], a tile-component of an original image or its lower resolution is decomposed into LL, HL, LH and HH subbands, where LL is the lower resolution used for further decomposing. Each subband includes many codeblocks. The individual bit-planes of the quantized DWT (Discrete Wavelet Transform) coefficients in a codeblock are coded in one of the three coding passes: significance propagation, magnitude refinement, and cleanup.

The basic content unit in a JPEG2000 codestream is packet which is associated with a specific tile, layer, component, resolution and precinct. Each layer consists of a number of consecutive bit-plane coding passes from each codeblock, including all subbands of all tile-components. Referring to Figure 1, a Codeblock Contribution to Packet (CCP) is explicitly signalled within the packet header, and CCPs in a packet body is arranged in the order of HL, LH and HH subbands. Additionally, an important restriction imposed by the JPEG2000 standard is that the last byte of a CCP can never be 0xFF [10].

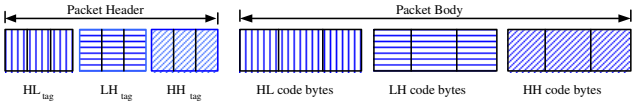

Fig. 1. Packet Structure: Packet bodies include CCPs (Codeblock Contribution to Packet).

3. THE COMPLIANT ENCRYPTION SCHEME

The goal of a compliant encryption is to output a protected codestream whose length is the same as that of the input codestream, and any word in the output content bitstream is not in the interval [0xFF90, 0xFFFF]. Our proposed scheme selects CCP as the processing unit because the end byte of CCP is non-0xFF. Assume that the encryption engine shares a secret key with the decryption engine in advance. The following subsections elaborate how to adapt stream ciphers to generate compliant ciphertexts.

3.1. Compliant Encryption

Figure 2 illustrates the process of encrypting a CCP **P** into a compliant bitstream **C**. Assume that the CCP has a unique identifier which is determined by tile, component, subband, etc, and $\mathbf{P} = \{P_1, P_2, \cdots, P_n\}$ is of *n* bytes. The encryption process proceeds as follows:

(1) Generate key stream **K** of *n* bytes from the secret key and CCP identifier with a stream cipher key generator.

(2) Input **P** into the Microblock encryptor.

(3) Compute $\mathbf{C} = \mathbf{P}+\mathbf{K} \bmod 256^n = \{C_1, C_2, \cdots, C_n\}$, where C_i is the i^{th} byte, $i = 1, 2, \cdots, n$.

(4) Check whether **C** is syntax compliant. Specifically, for any $i \in [1, n-1]$, if $C_iC_{i+1} \in$ [0xFF90,0xFFFF], **C** is non-compliant. In this case, let $\mathbf{P} \leftarrow \mathbf{C}$, and go back to step (2).

(5) Check the last byte C_n. If C_n = 0xFF, **C** is non-compliant, let $\mathbf{P} \leftarrow \mathbf{C}$, and go back to step (2).

(6) Output **C** as the compliant bitstream.

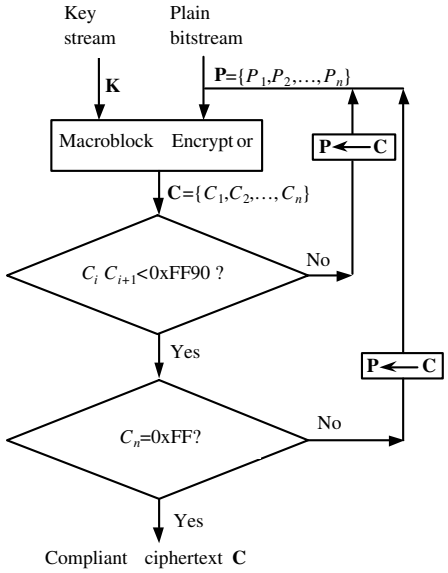

Fig. 2. Diagram of Compliant Encryption.

3.2. Compliant Decryption

The decryptor parses a protected codestream into CCPs as the encryptor does. For each CCP, the compliant decryption is illustrated in Figure 3.

(1) Generate the key **K** of *n* bytes from the shared key and CCP identifier using a stream cipher key generator.

(2) Input bitstream **C** into the Microblock decryptor.

(3) Compute bitstream $\mathbf{P} = \mathbf{C} - \mathbf{K} \bmod 256^n = \{P_1, P_2, \cdots, P_n\}$, where P_i is the i^{th} byte, $i = 1, 2, \cdots, n$.

(4) Check whether **P** is syntax compliant. Specifically, for any $i \in [1, n-1]$, if $P_i P_{i+1} \in$ [0xFF90, 0xFFFF], $(i = 1, 2, \cdots, n-1)$, **P** is non-compliant, let **C** $\leftarrow$ **P**, and go to step (2).

(5) Check the last byte P_n. If P_n = 0xFF, **P** is non-compliant, let **C** $\leftarrow$ **P**, and go to step (2).

(6) Output **P** as the plaintext bitstream.

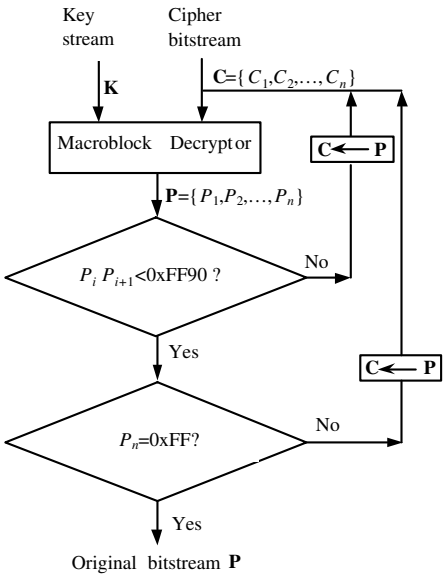

Fig. 3. Diagram of Compliant Decryption.

4. PERFORMANCE AND ANALYSIS

4.1. Rationale

In the present scheme, CCP is encrypted recursively until the ciphertext meets two requirements:

- any word (two contiguous bytes) is within [0, 0xFF90)
- the end byte is not 0xFF.

That is to say, a protected bitstream looks like a CCP. Obviously, this kind of protected bitstream is uniquely determined with the minimal number of encryptions. Thus, the encryption process is reversible. On the other hand, any CCP can be encrypted into a compliant bitstream.

4.2. Scalability

Because the CCP is the processing unit, the packets in a protected codestream can be re-organized with any progressive orders. The protected codestream can also be processed at the granularity of codeblocks without going through any encoding again. Therefore, the present scheme maintains the scalability of the original JPEG2000 codestream.

4.3. Error Resilience

In a JPEG2000 codestream, an error in one coding pass will propagate to the rest of the coding pass bitstream. In the present scheme, because the processing is CCP oriented, an error is limited to the damaged CCP only. Thus, the present scheme maintains the error-resilience of JPEG2000.

4.4. Computational Cost

Because encryption is regarded as a randomization operation, the probability *p* of generating a compliant bitstream with one macroblcok encryption is

$$p > p_0^n = (\frac{255}{256})^n,$$

where p_0 is the probability that one byte is not 0xFF. The solid line in Figure 4 illustrates the probability of generating a compliant bitstream. Let $q = 1 - p$. Because the rounds of encryption are independent, the expect number of encryption for a compliant bistream is

$$E_n = p + 2qp + 3q^2p + \cdots = \sum_{r=0}^{+\infty}(r+1)q^r p = \frac{1}{p}$$

The dashed line in Fig. 4 shows the upper boundary of E_n.

The JPEG2000 standard defines the size of a code-block as no more than 4096 bits per bitplane. Thus the maximum size of a CCP is 4096α bits where α is the number of bitplanes. In fact, the size of a CCP is usually small due to the underlying efficient compression algorithm and layer segmentation. Because encryption operation is merely arithmetic module addition, and the key stream is generated only once, the computational cost is very small.

4.5. Variant

Although stream ciphers are exemplified in the above microblock encryption/decryption, block ciphers are applicable too. Technically, assume the size of employed block cipher is *b*, and the size of bitstream **P** is *n*, then ECB mode (chapter 9 of [11]) of the block cipher is selected for encrypting the first $b \times \lfloor n/b \rfloor$ bytes, and the aforementioned stream cipher is selected for encrypting the rest of the bitstream **P**.

5. SIMULATIONS

The sizes of CCPs vary from codeblocks, packets, and images. In this simulation, 5000 compliant bitstreams are used as test examples. The key stream is generated using RC4 (Chapter 17 of [11]) . The number of repeated encryption (arithmetic addition module) is shown in Table 1. In Table 1, the first column is the size of CCP in bytes, the other

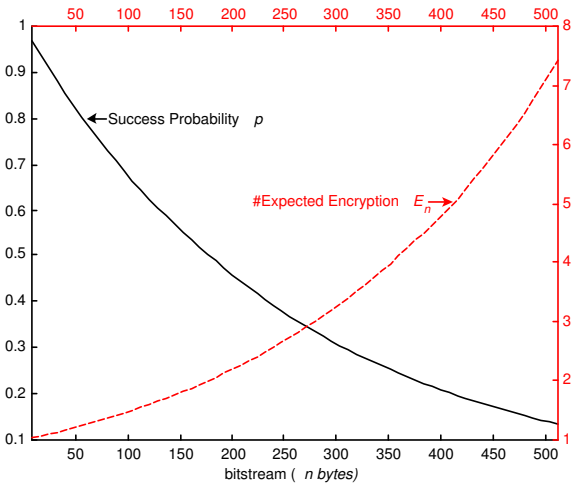

Fig. 4. Solid line: Success probability (lower boundary) of Macroblock encryption vs. bitstream size. Dash line: the number (upper boundary) of expected Macroblock encryption vs. bitstream size.

columns represent the percentage of repetitions for generating protected compliant bitstreams. Table 1 indicates that the small number of repetitions is of high probability.

Table 1. Percentage of encryption for generating a compliant bitstream.

CCP size	1	2	3	4	>4
8	96.5	0.1	3.4	0.0	0.0
58	69.6	22.1	7.5	0.8	0.1
108	65.3	23.0	9.9	1.5	0.3
158	61.2	25.6	10.8	1.9	0.5
208	55.9	29.0	10.9	3.2	1.0
258	52.1	30.6	11.7	3.9	1.7
308	49.3	31.7	12.7	4.5	1.9
358	48.0	29.1	9.8	8.3	4.8
408	44.8	29.3	10.6	6.6	8.6
458	43.0	28.2	11.8	7.4	9.6

6. CONCLUSION

JPSEC focuses on the security aspect of JPEG2000 codestreams and files. Encryption is one of its main concerns. However, straightforward encryption may generate superfluous markers in the protected codestreams. The compliant encryption method proposed in this paper overcomes the problem using the standard ciphers and keeps the structure of the codestream. The proposed method also provides full protection of the codestream, maintains all the nice properties of the original JPEG2000 codestream, such as error resilience and scalability. We believe that the basic principle presented here can be applied to encrypt other data formats, such as MPEG.

7. REFERENCES

[1] "Information Technology - JPEG 2000 Image Coding System", *ISO/IEC International Standard 15444-1*, ITU Recommendation T.800, 2000

[2] Jiangtao Wen, M. Severa, Wenjun Zeng, M.H. Luttrell and Weiyin Jin, "A Format-compliant Configurable Encryption Framework for Access Control of Video," *IEEE Transactions on Circuits and Systems for Video Technology*, 12(6):558-565, 2002

[3] Wenjun Zeng, Jiangtao Wen, M. Severa, "Fast Self-synchronous Content Scrambling by Spatially Shuffling Codewords of Compressed Bitstreams," ICIP, pp. III 169-172, 2002.

[4] Min Wu and Yinian Mao, "Communication-friendly Encryption of Multimedia," IEEE Workshop on Multimedia Signal Processing, pp. 292-295, 2002

[5] Vania Conan, Yulen Sadourny and Stève Thomann, "Symmetric Block Cipher Based Protection: Contribution to JPSEC," ISO/IEC JTC 1/SC 29/WG1 N2771, Oct. 2003

[6] Yongdong Wu, Robert Deng and Di Ma, "ImAccess:A Method for JPEG 2000 Access Control," Presentation on 29^{th} ISO/IEC JTC 1/SC 29/WG1 meeting, Seoul, Mar. 2003,

[7] Canon inc. "Encryption Tool for JPEG2000 Access Control," ISO/IEC JTC 1/SC 29/WG1 N2893, 2003

[8] Di Ma, Yongdong Wu and Robert Deng, "Analysis of Canon Encryption Scheme," Communications in JPEG2000 Security group (JPSEC), Nov. 14, 2003

[9] Hongjun Wu and Di Ma, "Efficient and Secure Encryption Schemes for JPEG2000," ICASSP 2004, pp. V869-872, see also ISO/IEC JTC 1/SC 29/WG1 N2937, 2003

[10] D. S. Taubman and M. W. Marcellin, *JPEG2000 - Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, pp.495, 2001.

[11] Bruce Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd edition, John Wiley & Sons, ISBN: 0471117099, 1995